International Journal of Research and Applications
April - June © 2014 Transactions; 1(2): 40-43
ISSN (online): 2349-0020
http://ijraonline.com
COMPUTERS ORIGINAL ARTICLE

Embedded Computing : Partitioning the Software between Clients and Servers

[1] Siruvoru Vahini , [2] Nampally Vijay Kumar
[1 2] Assistant Professor, Department of Computer Science & Engineering ,
[1 2] SR Engineering College (Autonomous), Affiliated to JNTU, Warangal - 506 371.
[1] vahini.siruvoru@gmail.com and [2] nampallyvijaykumar@gmail.com

ABSTRACT

The energy efficiency of mobile applications has been a highly tackled research problem within the last years. Many research groups have focused on optimizing the hardware of mobile devices, as well as their middleware and applications, increasing both the devices' uptime and their users' satisfaction. However, only scare work has analyzed to partition the application between client and server. In this paper, we focus on social games which are played by several players connected to a server via their mobile devices. Current state of the art is to keep everything on servers and synchronize the state across all clients which makes real-time media oriented computation can perform poorly on servers due to varying loads, and can result in interruptions in execution. Due to lots of jobs, sometimes the computation does not scale with number of users on servers. Thus in this paper, we propose to partition the application between embedded device and server using Android for performance benefit and energy saving.

Keywords: Energy efficiency, embedded devices, Server, Social-games, Android, Partitioning.

I. INTRODUCTION

In the last years, the energy efficiency of mobile applications has been claimed to be of topmost importance and many research groups have focused on energy-optimizing the hardware of mobile devices, as well as their middleware and applications [1]–[3]. Besides increasing device uptimes, one of the major motivations behind all these efforts is to increase the user experience for mobile applications. However, scarce work has focused on the frequency of energy-efficiency issues in mobile applications and their influence on user satisfaction [4], [5].

The remainder of this paper is structured as follows: Sect. II presents background which includes purpose of the research and related work focusing on user satisfaction and feedback w.r.t. the energy efficiency of mobile apps. Following, in Sect. III we describe the discussion of client server systems in the present scenario. Afterwards, Sect. IV gives the methodology of the research. Sect. V concludes this paper.

II. BACK GROUND

A) Purpose of Research

This work will result in:

(a) Techniques for partitioning of an Interactive client server game

(b) Infrastructure for communication and event processing using distributed processing – client-server based approach.

(c) Experimental system for partitioning on client server platforms for performance benefit and energy saving.

B) Related Work

To the best of our knowledge, only scarce work exists that focuses on the frequency of energy-efficiency issues in mobile applications and their influence onto user satisfaction. For the analysis presented in [4], Pathak et al. crawled several online forums and bug trackers relating to mobile platforms and their applications, searching for energy efficiency issues. They analyzed posts relating to energy problems and derived taxonomy for faulty implementations and hardware errors they classified as different types of *energy bugs*. In contrast to our work, Pathak et al. did not analyze the distribution of energy bugs onto individual applications. Neither had they analyzed the influence of energy bugs onto user satisfaction and their behaviour (e.g., the negative grading of applications due to user dissatisfaction). Heikkinen et al. [5] conducted two online questionnaires as well as two usage monitoring studies evaluating the influence of usage behaviour on energy consumption of mobile applications. Their findings revealed that users are interested in energy consumption statistics of their mobile devices as well as in information on how they could optimize their devices.

However, Heikkinen et al. did not evaluate whether or not high energy consumption of a specific application is recognized by users and thus, leads to dissatisfaction.

III. DISCUSSIONS

Embedded devices are becoming very powerful; more and more functionality is becoming available on the handheld mobile devices. Some embedded end devices such as Smartphone's, Tablets, notebooks or net books have powerful processors such as the dual cores, GPUs, DSPs and also have large amount of memory as well based on the flash or SSD (solid state device) disks. Client server systems have been the norm even for embedded applications. Currently, most of the part of an application uses a server where majority of the processing takes place. But such an approach has many disadvantages:

- Communication with the server can be sporadic with sometimes bad connectivity and so some applications don't work at all when the signal is weak.

- Communication with server costs a lot of energy – in general data movement and communication takes a lot of energy than the computation

- The same computation on server is very energy in-efficient since new processes have to be spawned; some middleware layers have to manage the client server interactions. In contrast, embedded devices are much more energy efficient by design; also operating systems like Android are quite lightweight and don't have layers.

- Real-time media oriented computation can perform poorly on servers due to varying loads, and can result in interruptions in execution. Due to lots of jobs, sometimes the computation does not scale with number of users on servers.

Due to above reasons, many times it may be a good idea to move certain type of computation out of the servers. However, such an approach has pros and cons:

Pros:

- Better system availability: even though the connectivity is poor, the system can function on the embedded device in a stand-alone manner.

- Better energy efficiency: Due to a lesser amount of communication, the application can function with better energy efficiency.

- Better responsiveness: For interactive multimedia and games, real time response is better.

Cons:

- Quality of computation: Since the computation executes locally on a device, resources may be lesser than server and application might have to be scaled down to execute within smaller number of local resources.

- Servers are backed up – thus, data losses are rare; if an embedded device fails, all the data is lost.

Due to the above tradeoffs, it is an interesting research problem to decide how application software should be partitioned between the embedded device and the server. For the purposes of this proposal, we focus on social games which are played by several players connected to a server via their mobile devices. Current state of the art is to keep everything on servers and synchronize the state across all clients and this is not a good idea due to above reasons. However, a given application is not arbitrarily partition able between a client and a server. Of course, certain services, data-bases and other utilities may only be available on servers and thus are not movable to end devices. For example, password verification is a server only activity.

However, during the execution of the application itself, one could have choices in deciding where to keep the global game state for example. Part of it can be kept locally and part globally on server. Android software platform which is open source provides excellent infrastructure for experimentation in this regard which we will use [8].

First we will determine which parts of an application are client only and which are server only. This is mainly determined by the support for functionality. For the remaining part of the application, we will first determine: which part of the state is to be kept local and which one to keep global ie, how to partition the remaining application? There are several issues faced in deciding the partitioning [6]-[7]. Data communication is one of them. The partitioning could be done to keep most of the data local so that communication is minimized. On the other hand, events are generated during the interactive application from multiple users and it may be best to accumulate them in a single server and then decide the action – which part of the state should be kept on server. Some state could even be duplicated in client and server so that the processing does not have to stall. But such a state duplication is to be minimized since it wastes resources. Using the results of the research, we will partition the social game on clients and servers and measure performance and energy efficiency.

IV. METHODOLOGY

The following are the broad objectives of this work:

(1) Generate a pool of client server interactive applications for embedded mobile devices based on Android operating system. Select a large interactive real time application such as social gaming.

(2) Partition application in terms of client or server only parts based on the functionality – ie, databases, services on server side, cameras, sensors etc. on the device side.

(3) For the remaining parts, determine the partition based on data communication for the client (mobile device) and for the server side.

(4) Perform application partitioning based on event processing and generator and receiver of events for the client and the server.

(5) Determine the minimal state that may be replicated on both client and server to improve performance and energy.

(6) Evaluate performance and energy of the partitioned application.

V. CONCLUSION

As outlined earlier many advantages exist to push things out of servers onto embedded clients. If this project is successful, it will tap into enormous power of many embedded devices unlike a centralized server. Servers are coupled with data centers and hosting servers and data centers is very expensive due to cooling needed. Energy efficient embedded devices on the other hand, can carry out the same computation in a more energy efficient manner. Thus, it makes sense to move computation out of servers. Further work can be done on how to support distributed cloud system using these devices.

REFERENCES

[1] K. Naik, "A Survey of Software Based Energy Saving Methodologies for Handheld Wireless Communication Devices," University of Waterloo, ON,CA, Technical Report 2010-11.

[2] Z. Zhuang, K. Kim, and J. Singh, "Improving energy efficiency of location sensing on smartphones," in *MobiSys '10*. ACM, 2010, pp. 315–330.

[3] A. Pathak, Y. C. Hu, and M. Zhang, "Where is the energy spent inside my app?: fine grained energy accounting on smartphones with Eprof," in *EuroSys '12*. ACM, 2012, pp. 29–42.

[4] A. Pathak, Y. Hu, and M. Zhang, "Bootstrapping energy debugging on smartphones: a first look at energy bugs in mobile devices," in *HotNets-X*. ACM, 2011, Article No. 5.

[5] M. V. Heikkinen, J. K. Nurminen, T. Smura, and H. H¨amm¨ainen, "Energy efficiency of mobile handsets: Measuring user attitudes and behavior," *Telematics and Informatics*, vol. 29, no. 4, pp. 387–399, 2012.

[6]. "Partitioning Web Applications Between the Server and the Client": Janne Kuuskeri, Tommi Mikkonen, Department of Software Systems, Tampere, University of Technology (2009).

[7]. S. Chong, J. Liu, A. C. Myers, X. Qi, K. Vikram, L. Zheng, and X. Zheng. "Secure Web Applications via Automatic Partitioning." In SOSP, 2007.

[8] Android on Mobile Devices: An Energy Perspective- Tapas Kumar Kundu and Kolin Paul,Dept. of CSE,IIT,Delhi 2010 10th IEEE International conference on Computer and Information Technology